Designing and implementing an smart monitoring and management system of environmental and electrical conditions of server rooms

University of Guilan

Mokadar Daemdoost, Amin
amindaemdoost@yahoo.com

Abbasi, Poorya
hey@pooryaa.com

30 Jan 2023

Abstract

This thesis presents the design and implementation of a server room monitoring system, aimed at tracking important environment variables such as temperature, humidity, and dust levels. The system is designed to provide real-time updates on the conditions within the server room, ensuring the proper functioning of the servers and other equipment.

For testing purposes a data generating system has been implemented in order to mimic the behavior pattern of an actual sensor. The data is then processed and stored using a backend system built with Node.js, Prisma, and PostgreSQL. This allows for efficient data management and analysis, enabling the system to quickly identify any issues and provide insight into the server room's overall performance. The data is also visualized using a user panel that has been implemented using React, Next.js, and Tailwind. The panel provides users with easy-to-understand visualizations of the data, allowing for quick identification of trends and patterns.

Overall, the monitoring system developed in this thesis provides a comprehensive solution for monitoring server room environments, with the potential to enhance performance, increase reliability, and reduce downtime. The system allows for the continuous tracking and analysis of critical environmental variables, providing users with valuable insight into the server room's overall performance. The visualizations provided by the user panel allow for quick identification of trends and patterns, enabling users to quickly identify and address any issues that may arise.

Contents

Chapter 1

Introduction

1.1 Backgrounds

The growing dependence on information technology has led to an increased need for server rooms in various industries. Server rooms house the critical infrastructure needed to support the growing demand for data storage and processing. However, maintaining a server room can be a complex and challenging task. The performance of servers can be affected by several factors, such as temperature, humidity, dust, water leakage, voltage, and current. These factors can cause damage to the servers, leading to downtime and loss of critical data. Therefore, it is crucial to monitor the server room's environment to ensure that it is operating efficiently and reliably.

In recent years, advancements in sensor technology have made it easier to collect data on environmental variables within the server room. The data collected can provide insight into the server room's performance, enabling quick identification of issues before they become critical. However, managing and analyzing the data collected from sensors can be a daunting task. Therefore, there is a need for a comprehensive monitoring system that can efficiently collect, manage, and analyze data on environmental variables within a server room. In the following sections some of these difficulties are going to be discussed.

1.1.1 Related Works

Based on the article[1], it was argued that two of the factors that needed to be monitored in a server room were temperature and humidity. Therefore, in this implementation a Wemos DHT Shield wireless sensor was utilized to collect this data so the users would be able to access the real-time data on a web server hosted on a raspberry pi. In addition to that there was a notification system implemented in this system so when certain events happened it would send a message to the defined users via whatsapp. For example, the system would notify the administrator when humidity reaches 60 percent.

In another implementation provided in [2], the fundamental parts of the hardware was DHT 11 sensor, arduino UNO and ESPRESSO LITE and the system was designed as the figure 1.1 shows. In this system, humidity and temperature of the environment was collected using an Arduino UNO and then was transferred to

the cloud using a ESPRESSO LITE which after that would be visualized using a Thingspeak platform.
This system was not implemented to work as a real time data collecting system and the arduino reads the data from DHT 11 every 2 seconds and if anything unordinary happens like temperature going above 35 ° C or humidity reaches 80% it would send the data to the cloud and it would trigger an alarm using the buzzer integrated into the system. but in ordinary situations it would only send these data every 15 seconds.

In [3], some factors were introduced to monitor events such as unauthorized access, temperature and humidity fluctuations, smoke and leaks, airflow and vibrations so suitable action would be done in case of fire, earthquake, etc. In addition some saturation should be considered like when there is no internet connection or there is no power so we could get the information about the server room even in these times. Some of the solutions for this situation in this article was using an UPS and activating it automatically when there is no electricity to run the system. Also some of the previous studies were discussed in this article that can be seen in figure 1.2. In those systems one of the most common ways to connect a number of sensors together wirelessly was to use a WSN system that is considered to have some issues and challenges, which includes limited resources (e.g., processing, memory and power, etc.), security attacks.
All being considered in this design a raspberry pi was used as a smart object that collects sensor data in the area and connects to the system using a LORA gateway. In this implementation, the number of smart objects can be increased that can help with the scalability of the system. Using these smart objects also helps with taking appropriate action in case of a certain event and it can alarm the staff, control the air conditioning system, control UPS in case of electricity outage and so on.

One of the interesting views that was considered in [4], was paying attention to cost when it comes to taking actions like lowering the temperature in a way that it wouldn't cost us unnecessary electricity usage hence it would be a considerable feature when it comes to the small businesses that needs to work as efficient as they can to maintain their sources. This system was designed for a tropical area like Thailand where the temperature is between 23 and 30 celsius degrees so in most companies using an air conditioner was needed to reduce the temperature to below 20 ° C in order to prevent overheating in the highest load times. But keeping the temperature this low is mostly because in those systems there was no real-time monitoring system on the temperature. In addition to that based on American Society of Heating Refrigerating and Air-Conditioning Engineers (ASHRAE), the temperature in a server room should be up to 27 ° C so In this system, a temperature sensor was placed in each rack and for implementing this system a Crossbow TelosB mote running on TinyOS 2.1 was used that is capable of collecting data about temperature, humidity and light, then these data was sent to a computer(base station) using a WSN system and serial protocol so that it could be analyzed by a Fuzzy logic and be sent to a web server. In this Fuzzy logic some policies were introduced to control the temperature of the room by manipulating the air conditioners in the room.

The article [5] describes a prototype IoT solution and associated platform that

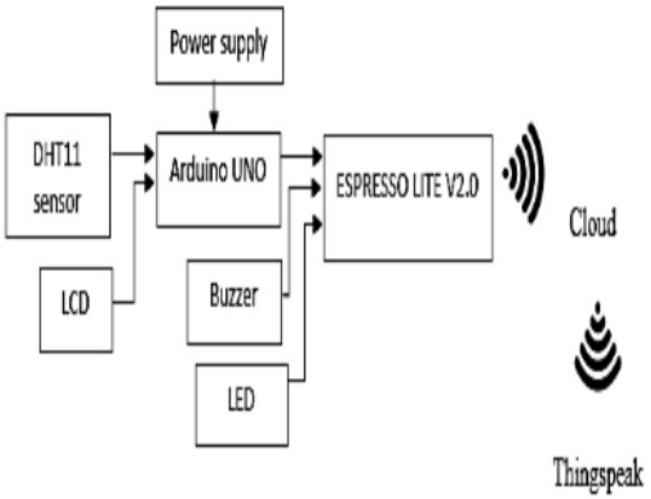


Figure 1.1: Designed system in article[2]

was applied to a data center monitoring system. The prototype uses SH31 instead of DHT11 which has more precise values rated at ± 0.3ºC, and does not require Wi-Fi which demands a gateway near and most likely a power source for it to last more than a few months. The text also mentions that the data center could be improved with CoolDoor by CoolDoor Pty Ltd, a refrigerated rack door where the air being inputted into the racks would get chilled before entry. This could improve energy consumption caused by the HVAC and present better temperature values. The developed system makes it possible to determine anomalies and prevent future problems. The experiment proved that these anomalies are much more frequent than initially thought and they can be caused by the hardware itself as much as the outdoor environment conditions. It was also possible to determine ways to improve the party's datacenter thermal flow by correctly managing the position of the server based on their 80 usage, since servers with expected higher usage should have more free space around them, providing better airflow and reducing the temperature gradient that represents one of the most important factors on hardware durability.

In this article[6] they design a temperature and humidity remote monitoring system and server room using Lattepanda and ThingSpeak to data cloud the monitoring result. The system design process consists of two parts, the first is the design of hardware reading data on the sensor. The second is sending data to ThingSpeak capable of being resuscitated which can be accessed via the internet using a browser. From the results of the test, it can be seen that Lattepanda can be used to monitor temperature and humidity in the server room. The temperature and humidity can be monitored in real time (every 30 seconds) through the channels provided by ThingSpeak.

1.1.2 Monitoring the environment

A monitoring system should be able to monitor the environment of the server room and alert the administrators in case of any abnormality. The following list contains

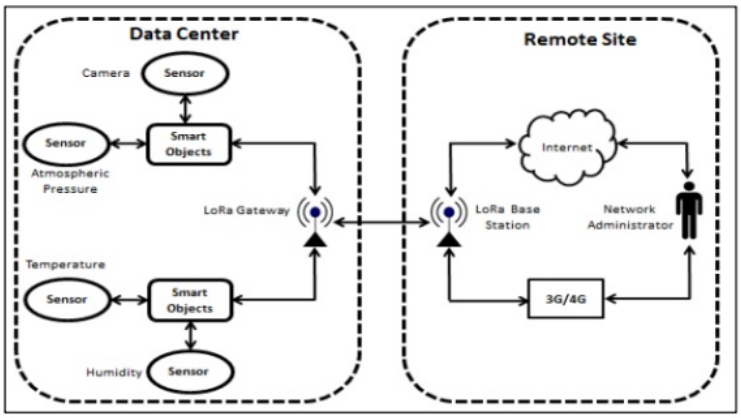


Figure 1.2: Designed system in article[3]

some of the most important parameters that should be monitored in a server room.

- Temperature

  Server room temperature should be maintained at a stable, suitable level to ensure the proper functioning of equipment. The recommended range of temperatures is between 18°C and 27°C [7].

  Too high temperature: If the temperature inside a server room rises above the recommended range, it can cause several problems. High temperatures can make equipment operate less efficiently and potentially fail altogether. High temperatures can also decrease the life expectancy of electronic devices and increase the chance that stored data will be corrupted [8].

  Too low temperature: If the temperature in a server room is allowed to fall below the recommended range, it can cause several issues. The cold air can lead to condensation—which leads directly to corrosion and equipment damage. In addition, low temperatures can decrease the efficiency of equipment and make it more likely to fail [8].

- Humidity

  On the other hand, High humidity levels can lead to condensation, which causes corrosion of the equipment and short circuits. High humidity levels can also create conditions that are conducive to the growth of mold and other microorganisms, which in turn damage equipment and affect indoor air quality [9].

- Dust

  Dust accumulation in a server room can degrade the performance and lifespan of equipment. Dust can block air vents, causing overheating, and it also attracts moisture leading to corrosion or other problems. So monitoring the levels of dust in a server room can help identifying and addressing potential problems.

The suitable range for dust density in a server room is around 0 to 3mg/m3.

- Water Leakage

  Serious consequences can result from water leakage in a server room, even if only a small amount of water leaks onto equipment. It is important to have proper monitoring systems and alerts that will notify personnel as soon as possible after any leak occurs [10].

- Electricity

  Monitoring the state of electricity, including voltage and current, is important in a server room to ensure the stability and reliability of the power supply to the equipment. Electrical voltage and current fluctuations can lead to problems with electronic equipment, such as data loss and corruption. In order to minimize these risks, server rooms are typically equipped with uninterruptible power supplies (UPS) and surge protection devices that help stabilize the voltage. The following list contains some common ranges for voltage and current [10].

  Voltage: The recommended operating range in from 208V to 240V and the maximum recommended limit is 264V.

  Current: The recommended operating range in from 20A to 40A (per phase)

- Movement

  Movement sensors, also known as motion detectors, can be used in order to detect unauthorized access to the room by detecting movements within the room and alerting the administrators.

- Smoke

  Smoke sensors can be used to detect smoke and alert the administrators in case of a fire.

- Authorization

  Authorization devices, such as cameras, fingerprints, keypads, etc., can be used to detect unauthorized access to the room by detecting the presence of a specific person.

1.1.3 Security

Another aspect of a monitoring system is that it should be able to prevent any intrusions and ensure that data remains intact and that access is restricted to only those with the appropriate privileges.

- Confidentiality And Authentication

in every IOT system, a proper authentication is crucial to ensure the security of the data so it wouldn't get into the wrong hands or get mutated by malicious actors and because the data is being transmitted remotely, more security measures has to be taken [11, 12].

- Authorization

  Security and reliability are the most important factors in our use case of network and other security measure implementations. There are trade offs like losing flexibility but these are the trade offs that we are willing to make to reach the maximum level of security possible [11, 12].

- Integrity

  In an IOT system, there is a risk of intruders attempting to impersonate legitimate users and alter or manipulate data. So it is important to implement verification and encryption methods to make sure in an event of a breach, the integrity of the data is still ensured and no nefarious actor in the system can mutate or access the data in any way because the consequences can be quite catastrophic [11, 12].

1.1.4 Real-time monitoring

Real-time monitoring is important in a monitoring system to ensure that the administrators are notified as soon as possible in case of any abnormality.

1.1.5 Data Visualization

Data visualization is an important part of a monitoring system, because it can provide a clear and compact representation of the data sent by the sensors. This allows administrators to quickly and easily identify trends, patterns, and anomalies in the server room environment, such as changes in temperature, humidity, power, and other environmental factors.
Data visualization can be achieved through a variety of means, such as graphs, charts, and maps. These visual representations can provide a lot of information in real-time, allowing administrators to make informed decisions about the state of their server room and respond quickly to any issues or problems. For example, graphs can be used to track trends over time.

1.2 Purpose

The purpose of this thesis is to demonstrate how an effective and efficient monitoring system can be implemented for server rooms using sensors, which are introduced in three categories in the tables 1.1, 1.2 and 1.3. The system will be designed to provide real-time data to a website, so that it can be monitored anywhere with internet access. This thesis will examine how this system was designed and implemented, including the selection of sensors and development of an online panel for monitoring data visualization.

Table 1.1: Electrical sensors

Name	**Measurement Unit**	**Output**
Voltage Sensor	Voltage	ADC
Current Sensor	Amper	ADC

Table 1.2: Environment sensors

Name	**Measurement Unit**	**Operating temp**	**Desired values**
Temperature sensor	Celsius	-20°C - 50°C	18°C - 27°C
Smoke sensor	ADC	-20°C - 50°C	-
Humidity sensor	Percent	-20°C - 50°C	40% - 60%
Water Leakage Sensor	-	-20°C - 50°C	-
Dust sensor	mg/m3	-20°C - 50°C	0 - 3

Table 1.3: Security sensors

Name
Movement sensor
Fingerprint sensor
Camera

Chapter 2

Data Generator Selection and Implementation

2.1 Introduction

In this implementation, in order to demonstrate how the monitoring system works instead of using actual sensors, a data generator has been used and it simulates reading of sensors (such as humidity, temperature and electricity).

2.2 Methodology

A data generator should have a random yet realistic pattern for effective system testing, so just using a simple random number would be insufficient. Therefore, in this thesis a controlled version of choosing a random number was used to mimic some parts of real-world criteria and avoid sudden changes. The methodology for this process can be seen in the accompanying flowchart 2.1.

2.3 Results

As discussed in the methodology section, the program was implemented, and the generated values tested with different parameters was saved and reported as line charts in the subsequent sections.

2.3.1 Generated temperature sensor data

- Parameters

  Starting temperature sensor Value: 22

  Minimum Possible temperature sensor value: 18

  Maximum Possible temperature sensor value: 27

  Maximum Possible Change in an iteration: 0.5

- Results can be seen in figure 2.2

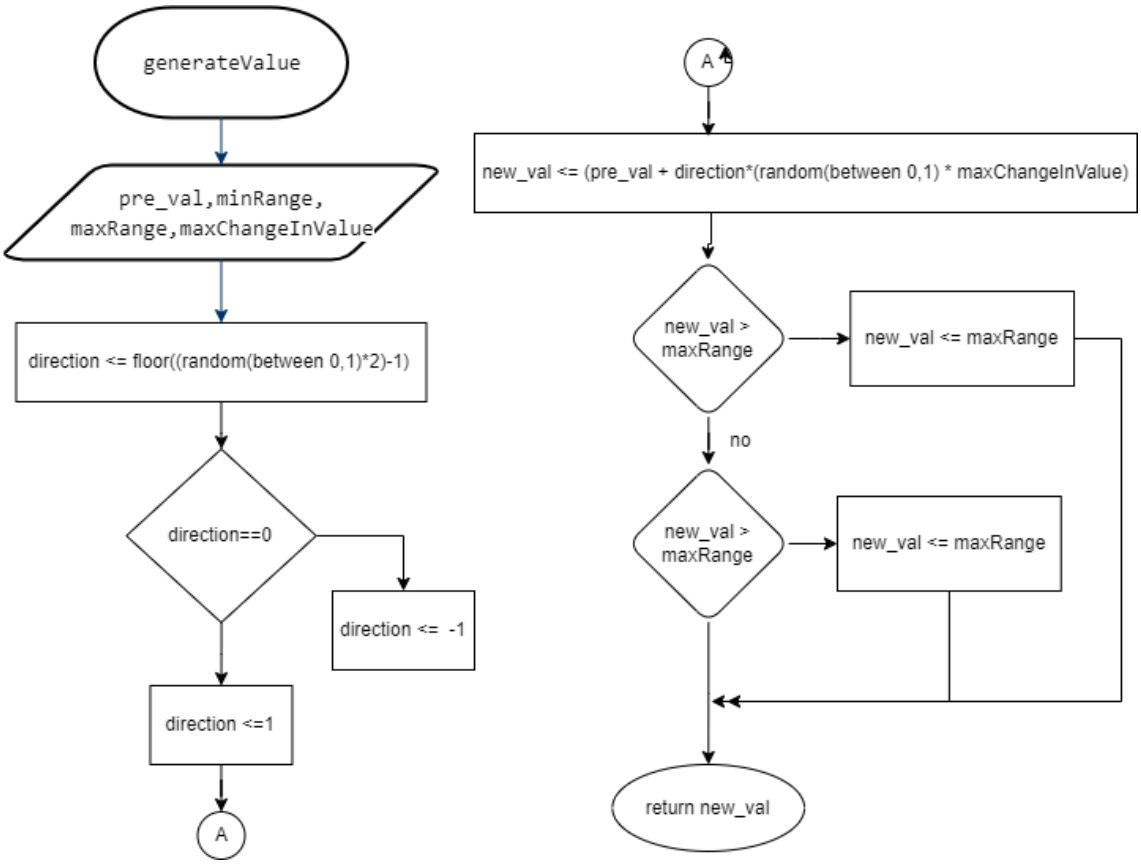
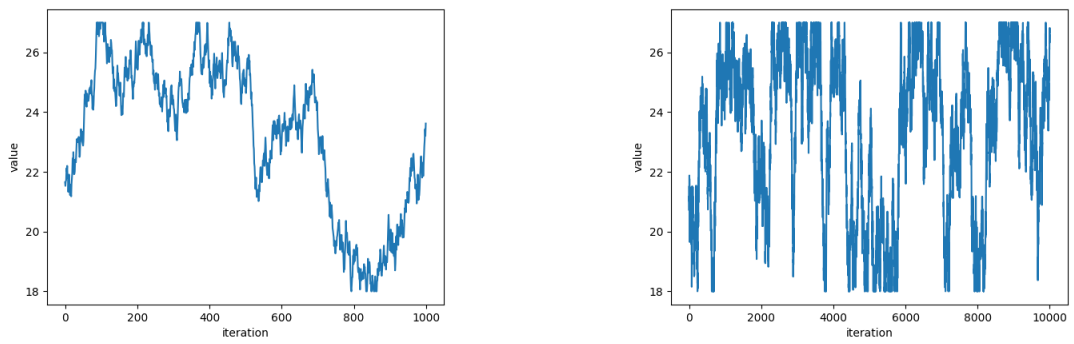
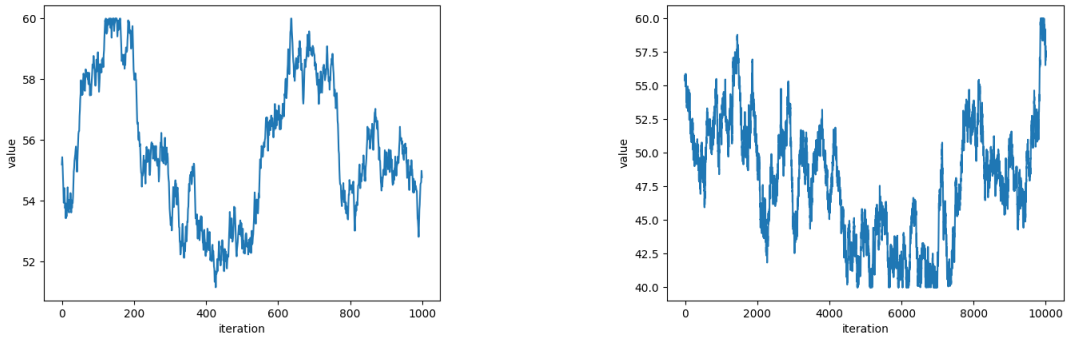


Figure 2.1: Flowchart of data generator



(a) For 1000 iterations (b) For 10000 iterations

Figure 2.2: Sample generated temperature sensor data



(a) For 1000 iterations

(b) For 10000 iterations

Figure 2.3: Sample generated humidity sensor data

2.3.2 Generated humidity sensor data

- Parameters

  Starting humidity sensor value: 55

  Minimum possible humidity sensor value: 40

  Maximum possible humidity: 60

  Maximum possible Change in an iteration: 0.5

- Results can be seen in figure 2.3

2.3.3 Generated dust sensor data

- Parameters

  Starting dust sensor Value: 1

  Minimum possible dust sensor value: 0

  Maximum possible dust sensor value: 3

  Maximum possible change in an iteration: 0.5

- Results can be seen in figure 2.4

2.3.4 Generated electricity voltage sensor data

- Parameters

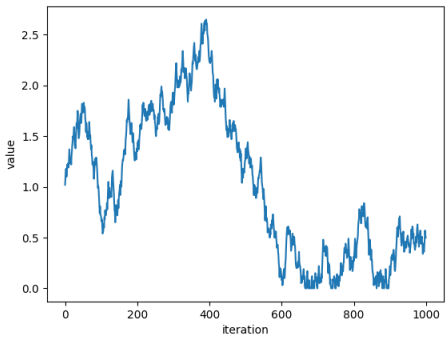
  Starting electricity voltage value: 220

  Minimum possible electricity voltage: 208

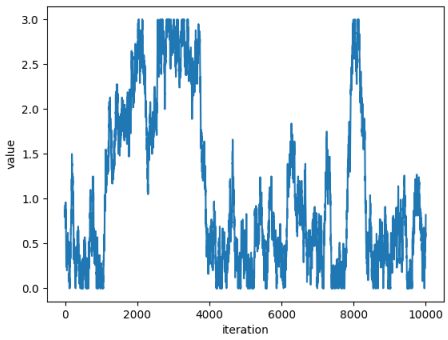
  Maximum possible electricity voltage: 264

  Maximum possible electricity change in an iteration: 1

- Results can be seen in figure 2.5

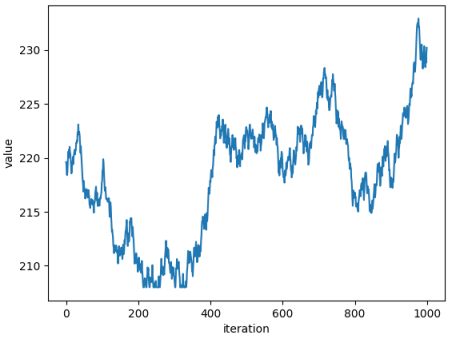


(a) For 1000 iterations

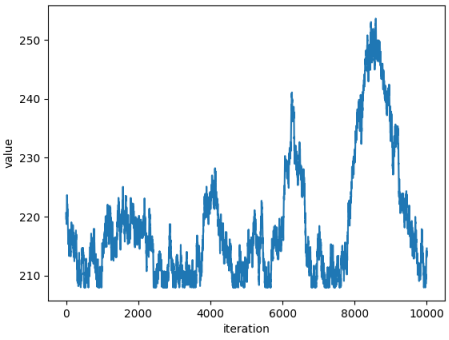


(b) For 10000 iterations

Figure 2.4: Sample generated dust sensor data

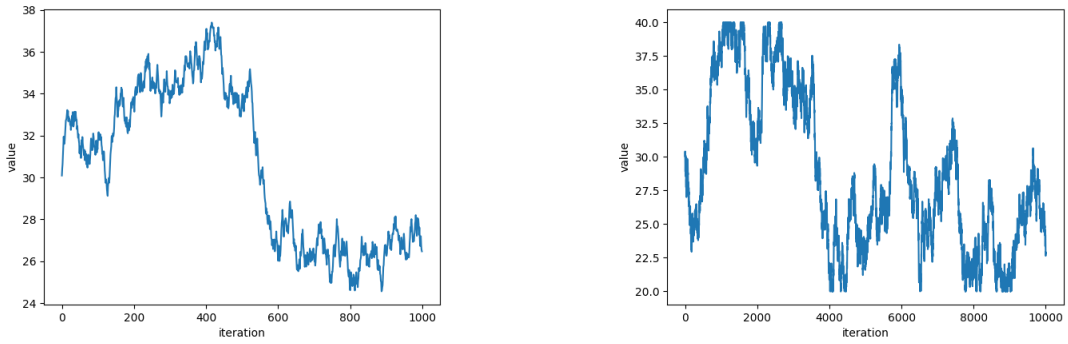


(a) For 1000 iterations



(b) For 10000 iterations

Figure 2.5: Sample generated voltage sensor data



(a) For 1000 iterations

(b) For 10000 iterations

Figure 2.6: Sample generated current sensor data

2.3.5 Generated electricity current sensor data

- Parameters

  Starting electricity current Value: 30

  Minimum possible electricity current: 20

  Maximum possible electricity current: 40

  Maximum possible change in an iteration: 1

- Results can be seen in figure 2.6

Chapter 3

Panel Design and Implementation

3.1 Introduction

A server room monitoring system panel is an essential component for ensuring the proper functioning of a server room environment. It provides real-time monitoring of key physical and environmental factors such as temperature, humidity, power, and network connectivity. The panel alerts administrators in case of any deviations from the defined thresholds, helping prevent potential data loss, hardware damage, and downtime. Additionally, this system provides historical data and reporting capabilities, allowing administrators to track trends and make informed decisions about capacity planning and infrastructure improvements. With the right server room monitoring system panel in place, organizations can ensure the safety and reliability of their critical IT assets.

3.2 Back-end

Node.js, PostgreSQL, and Prisma can be used together to build a robust and scalable back-end for a server room monitoring panel.

3.2.1 Node js

Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine, and is a critical component in building a server room monitoring panel's back-end. Node.js allows developers to use JavaScript on the server-side, providing a unified language for both client-side and server-side development. This helps to simplify the development process and increase developer productivity.
In a server room monitoring panel's back-end, Node.js can be used to create APIs or other server-side components that provide data for the monitoring panel. For example, it can be used to create a REST API that retrieves data from a database or other source and returns it in a format that can be consumed by the monitoring panel. Node.js is also well-suited for building fast and scalable applications, making it a good choice for a server room monitoring panel's back-end. Its asynchronous, non-blocking I/O model ensures that the back-end can handle a large number of concurrent requests without becoming slow or unresponsive [13].

3.2.2 Postgresql

PostgreSQL is an open-source relational database management system (RDBMS) and is a critical component in building the back-end for a server room monitoring panel. PostgreSQL is a powerful and reliable database system that can be used to store the data collected by the monitoring panel, such as temperature, humidity, power, and network connectivity. It provides a range of features for managing and querying data, including support for SQL, transactions, and advanced data types. The use of a relational database, such as PostgreSQL, is important for a server room monitoring panel because it ensures the data is stored in a structured manner. This makes it easier to retrieve and analyze the data, and also provides a robust solution for managing the data over time, ensuring its reliability and longevity. PostgreSQL also provides security features, such as user authentication and access control, to ensure that the data stored in the database is protected. This is especially important for a server room monitoring panel where sensitive information, such as server uptime, may be stored [14].

3.2.3 Prisma

Prisma is a modern database toolkit that provides a high-level API for working with databases and is a critical component in building the back-end for a server room monitoring panel. Prisma allows developers to work with databases, such as PostgreSQL, in a type-safe and efficient manner. It provides a set of TypeScript types for the data stored in the database, making it easy to catch bugs and errors at compile time. This helps to ensure the quality and reliability of the data stored in the database.
Prisma also provides a query builder for constructing and executing SQL queries, allowing developers to retrieve the data necessary for the monitoring panel. This makes it easy to retrieve and manipulate the data stored in the database, while also providing the performance and efficiency necessary for a server room monitoring panel's back-end. Prisma also provides a streamlined and modern API for working with databases, making it easier for developers to build robust and scalable back-ends. This can help to reduce development time and increase developer productivity [15].

3.3 Front-end

3.3.1 React

React is a JavaScript library for building user interfaces, and is a critical component in building a server room monitoring panel. React allows for the creation of reusable components, making it easier to manage and maintain the codebase as the application grows in complexity. This helps to ensure that the monitoring panel is both scalable and flexible, allowing for new features and functionality to be added as needed.
In a server room monitoring panel, React can be used to create components for displaying real-time data, such as temperature, humidity, power, and network connectivity. This data can be fetched from APIs or other sources and dynamically

updated using React's state management. This helps to ensure that the monitoring panel provides an up-to-date and accurate representation of the server room environment at all times. React's hooks and context also make it easy to add alerts and notifications to the monitoring panel, ensuring that administrators are promptly notified of any deviations from the defined thresholds. This helps to prevent potential data loss, hardware damage, and downtime [16].

3.3.2 Next js

Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine, and is a critical component in building a server room monitoring panel. Node.js allows developers to use JavaScript on the server-side, providing a unified language for both client-side and server-side development. This helps to simplify the development process and increase developer productivity.
In a server room monitoring panel, Node.js can be used to create APIs or other server-side components that provide data for the monitoring panel. For example, it can be used to create a REST API that retrieves data from a database or other source and returns it in a format that can be consumed by the monitoring panel.
Node.js is also well-suited for building fast and scalable applications, making it an appropriate choice for a server room monitoring panel. Its asynchronous, non-blocking I/O model ensures that the monitoring panel can handle a large number of concurrent requests without becoming slow or unresponsive [17].

3.3.3 Tailwind

Tailwind is a utility-first CSS framework, and is a critical component in building a server room monitoring panel. Tailwind provides a set of pre-defined CSS classes for quickly building responsive and customizable designs. This helps to significantly speed up the development process, as developers do not need to write custom CSS for common design elements.
In a server room monitoring panel, Tailwind can be used to style the components created with React. It provides a wide range of classes for controlling layout, typography, colors, and more, making it easy to create a consistent and professional look and feel for the monitoring panel. Tailwind also provides a set of responsive design classes, allowing the monitoring panel to adjust its layout and appearance based on the size of the user's screen. This helps to ensure that the monitoring panel is usable and accessible on a variety of devices, including desktop computers, laptops, tablets, and smartphones [18].

3.4 Panel overview

3.4.1 Dashboard

The dashboard is the main page of the monitoring panel, and provides a high-level overview of the server room environment. In this state of the panel as it can be seen in the image 3.1, online sensors and their data distribution has been reported.

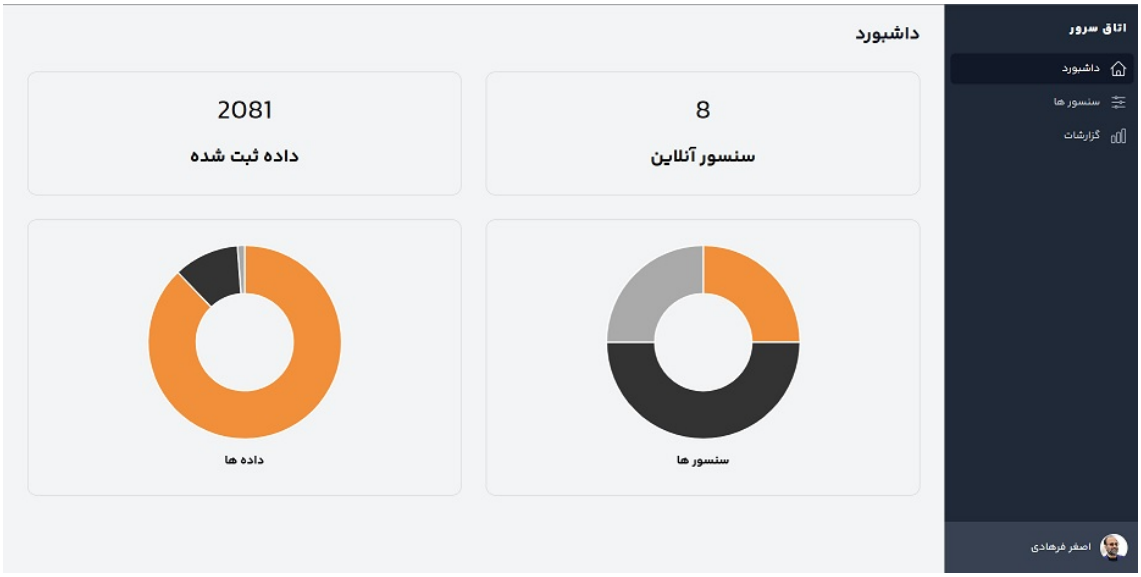


Figure 3.1: Dashboard tab in the panel

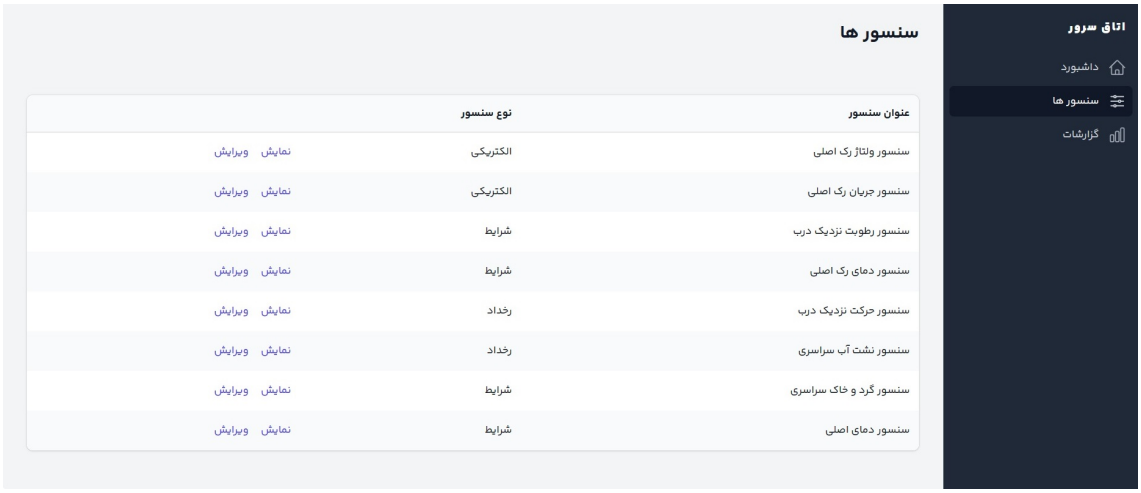


Figure 3.2: Sensors tab in the panel

3.4.2 Sensors

The sensors overview page provides a detailed view of the sensors in the server room and as it can be seen in the image 3.2, user can edit the sensor information or view the sensor data.

view sensor data

The sensor data page provides a detailed view of the sensor data and as it can be seen in the images 3.3, 3.4 and 3.5, user can view the sensor data in a graph.

sensor settings

In the sensor settings page, user can edit the sensor information and as it can be seen in the image 3.6, user can edit the sensor's name.

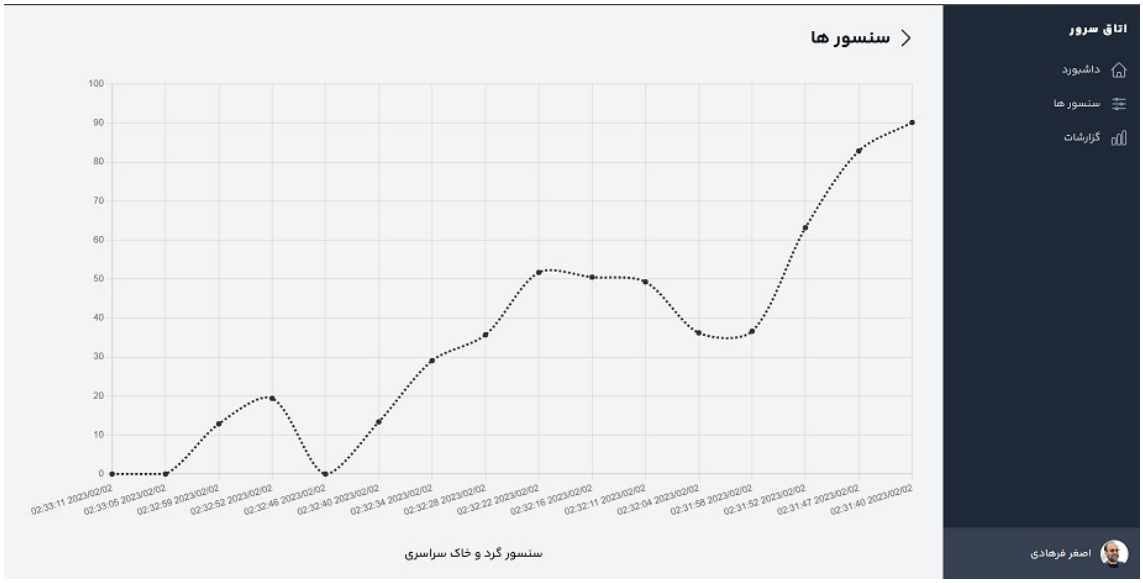


Figure 3.3: A dust sensor data in the panel

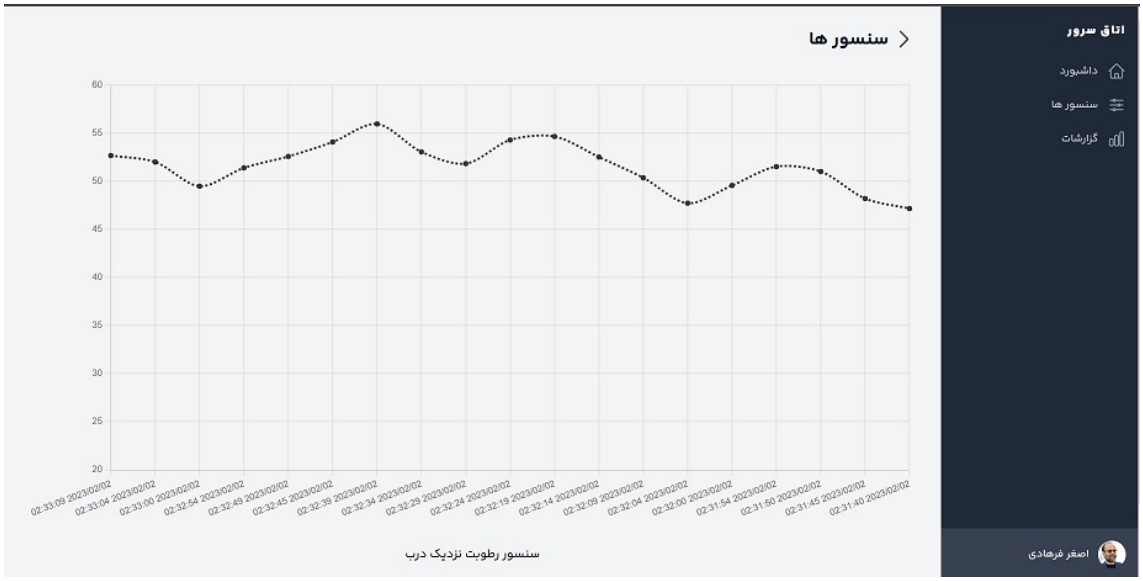


Figure 3.4: Humidity tab in the panel

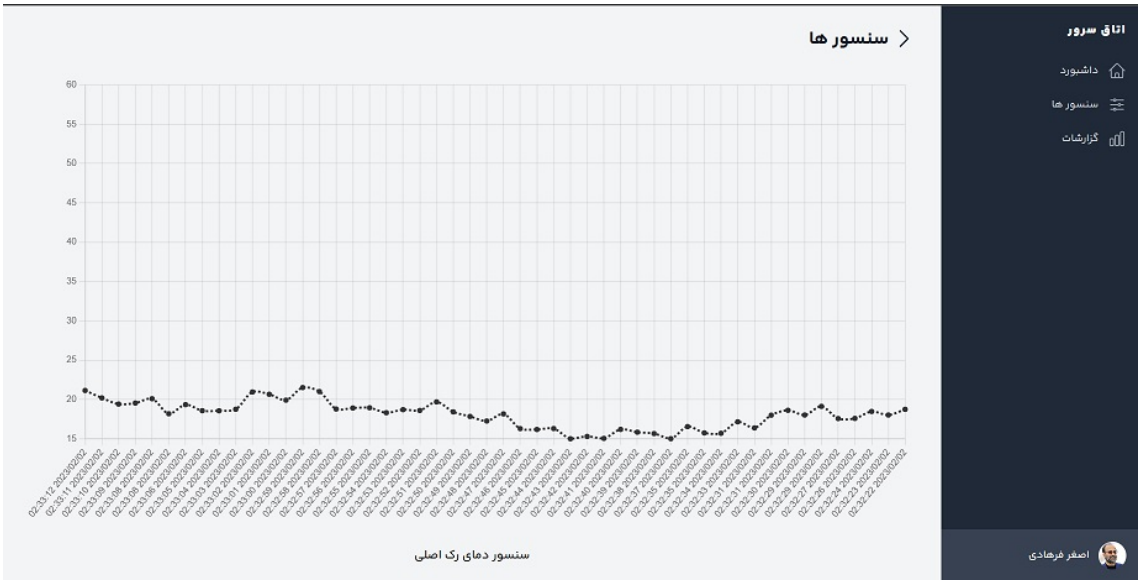


Figure 3.5: Temperature tab in the panel

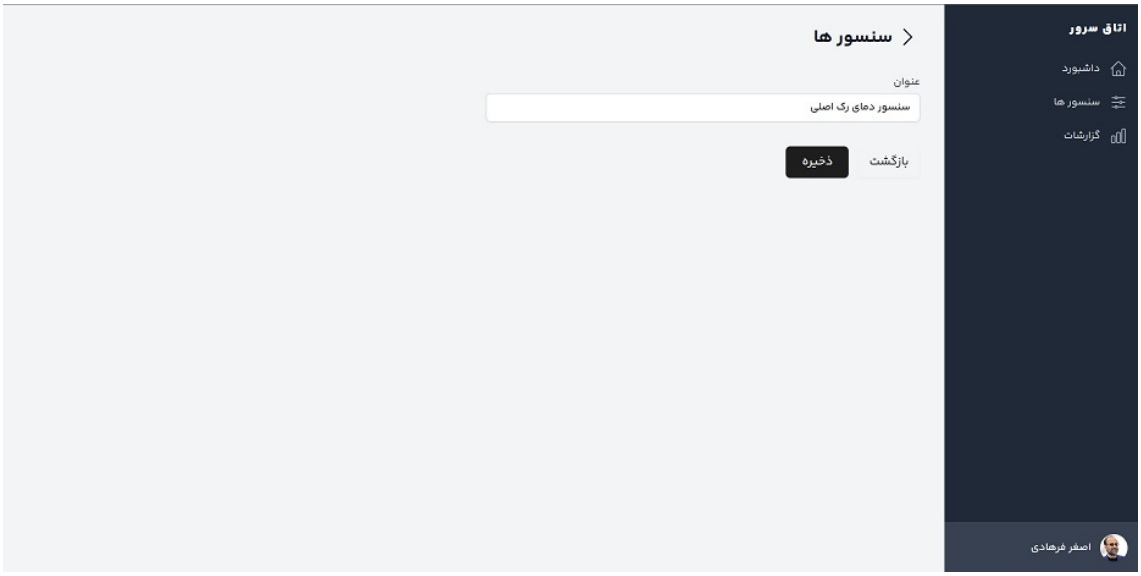


Figure 3.6: Sensor settings tab in the panel

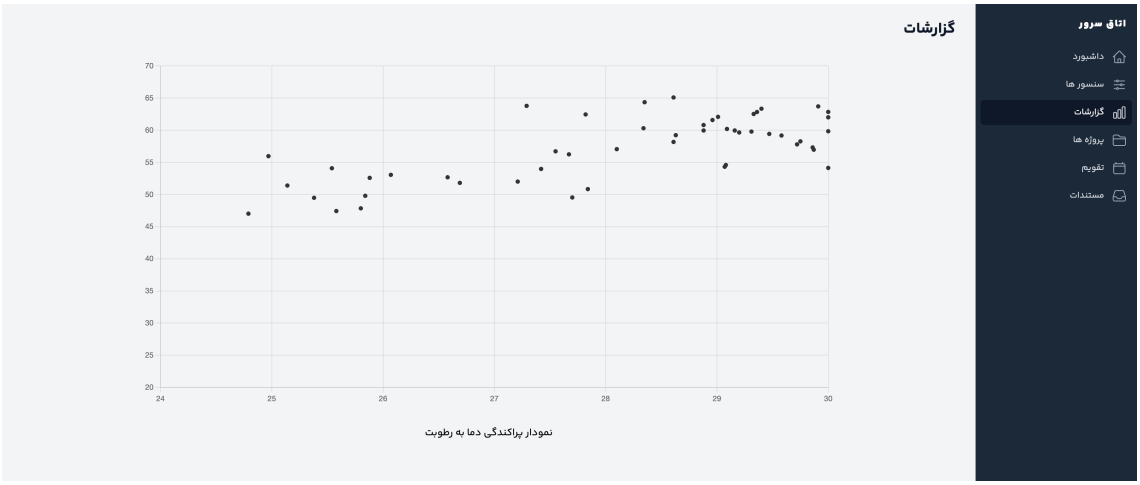


Figure 3.7: Relation between recorded data of temperature and humidity sensors

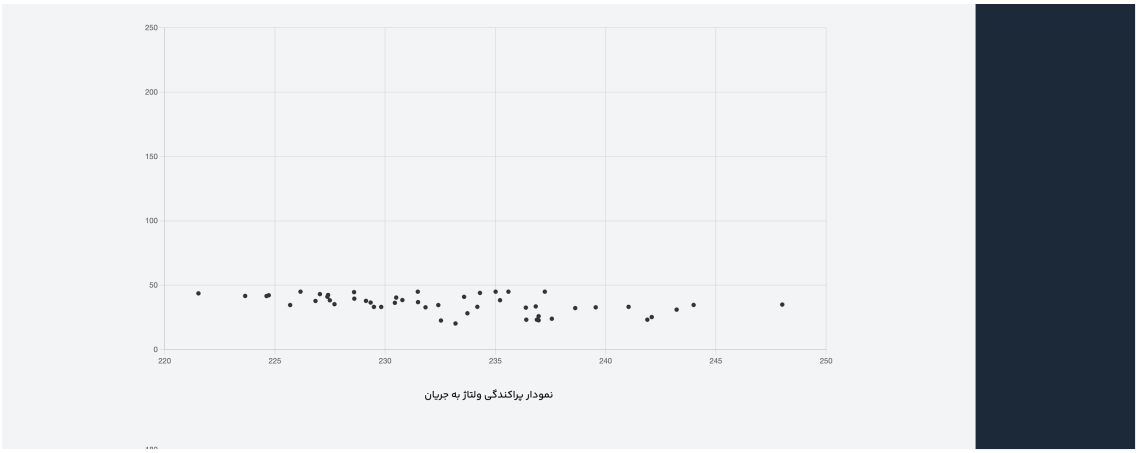


Figure 3.8: Relation between recorded data of voltage and current sensors

3.4.3 Reports

Scatter plots can be an effective way to visualize sensor data because they can help identify relationships or correlations between different sensor readings. In a monitoring system for a server room, multiple sensors are collecting data on various environmental variables, such as temperature, humidity, dust, and so on. By plotting the data collected from different sensors on a scatter plot, it may be possible to identify patterns or correlations between different variables. For example, you might observe that temperature tends to rise as humidity increases, or that there is a relationship between dust levels and humidity changes of the environment. So as it can be seen in figures 3.7, 3.8 and 3.9, the relation between temperature and humidity, voltage and current and also humidity and dust has been illustrated in scatter plots.

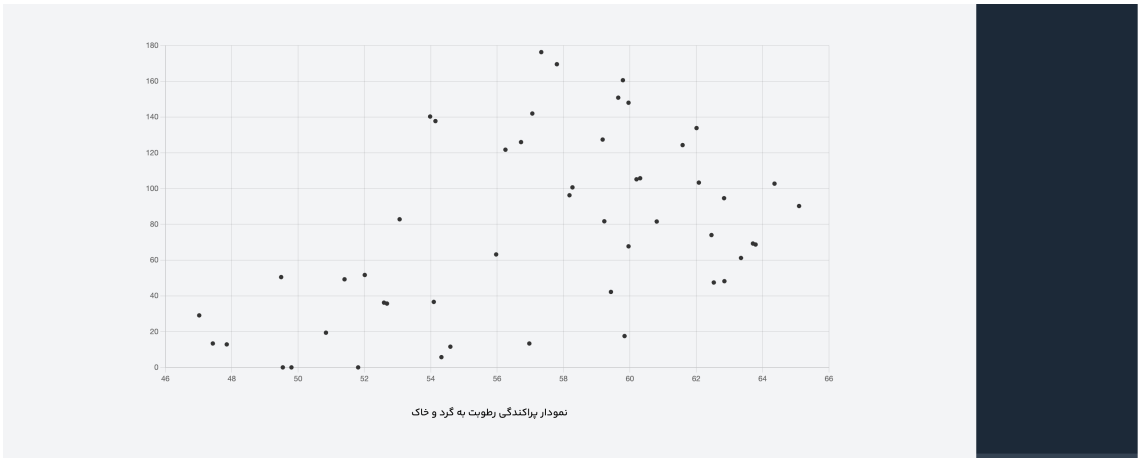


Figure 3.9: Relation between recorded data of humidity and dust sensors

Bibliography

[1] D. E. Kurniawan, M. Iqbal, J. Friadi, R. I. Borman, and R. Rinaldi, "Smart monitoring temperature and humidity of the room server using raspberry pi and whatsapp notifications," in *Journal of Physics: Conference Series*, vol. 1351, p. 012006, IOP Publishing, 2019.

[2] M. H. Zohari, V. Bala, and A. S. Abd Ghafar, "Server monitoring based on iot using thingspeak," *Journal of Electrical Power and Electronic Systems*, vol. 1, no. 2, 2019.

[3] W. Z. Khan, M. Aalsalem, H. Zangoti, M. Zahid, and M. K. Afzal, "Internet of things based physical and environmental monitoring system for data centers," in *2018 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET)*, pp. 41–44, IEEE, 2018.

[4] S. Choochaisri, V. Niennattrakul, S. Jenjaturong, C. Intanagonwiwat, and C. A. Ratanamahatana, "Senvm: Server environment monitoring and controlling system for a small data center using wireless sensor network," *arXiv preprint arXiv:1105.6160*, 2011.

[5] D. Santos, B. Mataloto, and J. C. Ferreira, "Data center environment monitoring system," in *Proceedings of the 2019 4th International Conference on Cloud Computing and Internet of Things*, pp. 75–81, 2019.

[6] T. Nasution, M. Muchtar, S. Seniman, and I. Siregar, "Monitoring temperature and humidity of server room using lattepanda and thingspeak," in *Journal of Physics: Conference Series*, vol. 1235, p. 012068, IOP Publishing, 2019.

[7] A. T. C. T. 9.9, "Data center storage equipment thermal guidelines, issues, and best practices." `https://resourcecenter.ashrae.org/File%20Library/Technical%20Resources/Bookstore/ASHRAE_Storage_White_Paper_2015.pdf`, 2015.

[8] "Data center cooling best practices." `https://www.datacenterknowledge.com/data-center-cooling-best-practices/`, Feb 2017.

[9] *Effect of Relative Humidity, Temperature and Gaseous and Particulate Contaminations on Information Technology Equipment Reliability*, vol. Volume 1: Thermal Management of *International Electronic Packaging Technical Conference and Exhibition*, 07 2015. V001T09A015.

[10] T. I. Association *et al.*, "Tia-942 telecommunications infrastructure standard for data centers," *Arlington Virginia: Telecommunications Industry Assn*, 2010.

[11] T. Saba, A. Rehman, T. Sadad, H. Kolivand, and S. A. Bahaj, "Anomaly-based intrusion detection system for iot networks through deep learning model," *Computers and Electrical Engineering*, vol. 99, p. 107810, 2022.

[12] W. Stallings, *Network Security Essentials: Applications and Standards*. William Stallings books on computer and data communications technology, Prentice Hall, third ed., 2007.

[13] Node.js, "Node.js." `https://nodejs.org/`.

[14] P. G. D. Group, "PostgreSQL." `https://www.postgresql.org/`, Feb. 2023.

[15] "Prisma | Next-generation ORM for Node.js & TypeScript." `https://www.prisma.io/`.

[16] "React – A JavaScript library for building user interfaces." `https://reactjs.org/`.

[17] "Next.js by Vercel - The React Framework." `https://nextjs.org`.

[18] "Tailwind CSS - Rapidly build modern websites without ever leaving your HTML.." `https://tailwindcss.com/`.